A1 been proposed and are disclosed in, e.g., Japanese Patent Application Nos. 9-32415, 9-32452, previously filed by the present applicant.--

Please replace the third full paragraph starting at line 19 on page 4 of the Specification with the following replacement paragraph:

A2 --Furthermore, analysis may be interrupted due to some cause other than memory shortage during program analysis. In such case, the analysis results previously obtained cannot be used, and program analysis must be redone from the beginning, resulting in poor analysis efficiency.--

Please replace the second full paragraph starting at line 19 on page 9 and continuing through line6, page 10 of the Specification with the following replacement paragraph:

A3 --In this embodiment, program analysis information contains a syntactic analysis tree (including a symbol table), call graph, flow graph, data flow information, program dependence graph, module I/O information, metrics information, redundancy information, maintenance document information, for example, as shown in Fig. 2. Of this information, the syntactic analysis tree, call graph, flow graph, data flow information, program dependence graph, module I/O information, for example, are generated by the program analysis information generation unit 11 shown in Fig. 1, while the metrics information, redundancy information, maintenance document information, for example, are generated by a program analysis unit 20_i, shown in Fig. 3, that analyzes a program by a batch process.--

Please replace the first full paragraph starting at line 7, page 10 of the Specification with the following replacement paragraph:

A4 --Note that the "symbol table" includes a group of information that represent meanings of symbols (variables) used in a program. For example, when character "a" is used at different locations in a program, they may indicate identical variables or different variables. For example, when variables expressed by an identical character are used in two different functions, if those

A4

variable are global variables, they are identical ones; if the variables are locally defined in the individual functions, they are different ones.--

Please replace the first and second paragraphs starting at line 7, page 12 of the Specification with the following replacement paragraphs:

A5

--The "data flow information" is information as a result of analysis of the data flow such as alias information of a pointer, definition-use chain information that indicates the use location of a variable defined at a given location, for example. Using this data flow information and similar information, for example, a data flow anomaly can be inspected. The data flow anomaly is a combination of illegal events with respect to data. All data must be used while keeping given rules, i.e., each data is used after it is defined, and is undefined finally. An illegal combination that does not keep such rules causes data flow anomaly.

Even when data flow anomaly is present, source code can be compiled. However, upon executing the compiled program in practice, data flow anomaly may appear when control takes an anomalous path. Hence, such anomaly is preferably removed in advance in the process of source code. Data flow anomaly detection is known in the art, as disclosed in Japanese Patent Application Nos. 9-32415 and 9-32452, by the present applicant.--

Please replace the third paragraph starting at line 15, page 13 of the Specification with the following replacement paragraph:

A6

--The "metrics information" pertains to numeration indices of software. The software analysis apparatus of this embodiment measures metrics which represent quantitative complexity, and those which represent qualitative complexity, and generates respective information. As metrics of quantitative complexity, two different kinds of quantities, i.e., a size metric that measures the physical description quantity of a program, and cyclomatic number that measures the complexity of a control structure are measured. On the other hand, module cohesion and module coupling relating to the contents of modules are measured, and are representative of the qualitative complexity of the software.--

Please replace the second paragraph starting at line 9, page 14 of the Specification with the following replacement paragraph:

A7

--The "maintenance document information" is a document group used upon maintaining a program. More specifically, this information describes lists of procedures, types, variable names defined in a program, and how the individual procedures are related, for example. This "maintenance document information" is used when a given programmer wants to understand a program created by another programmer. By acquiring the "maintenance document information", information can be provided in the hypertext format, and is much more convenient compared to paper-based documents.--

Please replace the second full paragraph starting at line 17, page 16 of the Specification with the following replacement paragraph:

A8

--By contrast, in the software analysis apparatus of this embodiment, the program analysis information storage unit 12 classifies program analysis information generated by the program analysis information generation unit 11 in units of kinds of analysis information in the respective layers, and sequentially stores them as a database on the information recording medium 13 such as a hard disk, for example.--

Please replace the first full paragraph starting at line 3, page 18 of the Specification with the following replacement paragraph:

A9

--Note that this embodiment uses as a database object-oriented database software, and generates program analysis information using an object-oriented language, e.g., C++, for example. Such database is used for the following reason. That is, since the program analysis information has a complicated structure and relation such as a graph structure, if a relational database is used, it is hard to store information. Also, use of such a database is convenient, since program analysis information generated as an object on C++ can be stored in a structure as it is generated.--

Please replace the paragraph starting at line 22, page 19 and continuing through line 6, page 20 of the Specification with the following replacement paragraph:

A10

--The program analysis unit 20_i executes an analysis process that can analyze by a batch process using various kinds of program analysis information stored in the information recording medium 13 by the program analysis information storage unit 12, and stores the process results as a batch analysis result file in the information recording medium 13 again. During this batch process, the operator need not input any instructions, and the apparatus of this embodiment automatically analyzes. The batch analysis result file contains, e.g., metrics information that pertains to the program scale, redundancy information, maintenance document information.--

Please replace the paragraph starting at line 3, page 22 of the Specification with the following replacement paragraph:

A11

--Other examples of the analysis process done by the program analysis units 20 are structure analysis, data flow anomaly analysis, for example. In this embodiment, the data flow anomaly analysis is done by an interactive process, but may be implemented by a batch process.--

Please replace the paragraph starting at line 18, page 24 and continuing through line 2, page 25 of the Specification with the following replacement paragraph:

A12

--A computer readable recording medium of the present invention will be described below. The software analysis apparatus of the present invention is implemented when the computer operates in accordance with a program stored in its ROM or RAM, for example. Alternatively, the software analysis apparatus may be implemented when a computer that loads an external program operates according to that program. Hence, when a computer mounts a recording medium that records such program, e.g., a floppy disk, CD-ROM, magnetooptical disk, magnetic tape, semiconductor memory, or the like, and loads the program therefrom, the present invention can be practiced on the computer.--